Low-Rank Approximations, Random Sampling and Subspace Iteration

Ming Gu, UC Berkeley
mgu@math.berkeley.edu

Modified from Talk in 2012
For RNLA study group, March 2015
Content

- Approaches for low-rank matrix approximations
- Random sampling and Subspace iteration
- Numerical experiments
- Future work
Low-Rank Matrix Approximation:

Problem Statement:
Given: mxn matrix A, and 0 < k < min(m,n) = n.
Goal: Compute a rank-k *approximation* to A.

- Fast low-rank matrix approximation is key to efficiency of superfast direct solvers for integral equations and many large sparse linear systems.
- Indispensable tool in mining large data sets.
- Randomized algorithms compute accurate truncated SVD.
- Minimum work and communication/Exceptionally high success rate.
Low-Rank Matrix Approximations: Current Approaches

Modified Gram-Schmidt with column pivoting.
(not reliable)

Rank-Revealing QR factorization.
(guaranteed but limited reliability)

Partial SVD.
(limited reliability)

The Lanczos Algorithm.
(too much communication)

Truncated SVD
(Best quality, but too slow)
Low-Rank Matrix Approximations: Current Approaches

Modified Gram-Schmidt with column pivoting.
(not reliable)

Rank-Revealing QR/LU factorization.
(Actually better than random sampling. Talks in April)

Partial SVD.
(limited reliability)

The Lanczos Algorithm.
(too much communication)

Truncated SVD
(Best quality, but too slow)
Low-Rank Matrix Approximations: Goals

Highly Efficient

Minimum communication

As accurate/reliable as Truncated SVD
Golden Standard: Truncated SVD

Given mxn matrix with \( n \leq m \), the SVD of \( A \) is

\[
A = U \Sigma V^T = (u_1 \cdots u_n) \begin{pmatrix} \sigma_1 & \cdots & \sigma_k \\ \vdots \\ \sigma_n \end{pmatrix} (v_1 \cdots v_n)^T
\]

The rank-k truncated SVD is

\[
A_k = U_k \Sigma_k V_k^T = (u_1 \cdots u_k) \begin{pmatrix} \sigma_1 & \cdots & \sigma_k \\ \vdots \\ \sigma_k \end{pmatrix} (v_1 \cdots v_k)^T
\]

Theorem [Can’t beat \( A_k \)] (Eckart & Young, 1936)

\[
\min_{\text{rank}(B) \leq k} \| A - B \|_2 = \| A - A_k \|_2 = \sigma_{k+1}
\]

\[
\min_{\text{rank}(B) \leq k} \| A - B \|_F = \| A - A_k \|_F = \sqrt{\sum_{j=k+1}^{n} \sigma_j^2}
\]
Low-rank Approximations: Strong Rank-Revealing QR

**Theorem** [Limited Warranty] (Gu & Eisenstat, 1994)

Given mxn matrix with n ≤ m, there exists a permutation Π,

\[
\begin{pmatrix}
R_{11} & R_{12} \\
R_{21} & R_{22}
\end{pmatrix} = Q \begin{pmatrix}
\sigma_1 & 0 \\
0 & \sigma_j
\end{pmatrix}
\]

with 

\[
1 \leq \frac{\sigma_j}{\sigma_j(R_{11})} \leq \sqrt{1 + 4k(n - k)}
\]

\[j = 1, \ldots, k.\]

- Factorization can be computed in O(mnk) operations.
- Basis for some popular low-rank matrix approximation schemes.
- Permutation can be arbitrary, excessive communication possible.
Low-rank Approximations: Strong Rank-Revealing QR

**Theorem** [Limited Warranty] (Gu & Eisenstat, 1994)
Given \( mxn \) matrix with \( n \leq m \), there exists a permutation \( \Pi \),

\[
A \Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ R_{22} \end{pmatrix} \quad \text{with} \quad 1 \leq \frac{\sigma_j}{\sigma_j(R_{11})} \leq \sqrt{1 + 4k(n-k)}

\]

\( j = 1, \cdots, k \).

- Factorization can be computed in \( O(mnk) \) operations.
- Basis for some popular low-rank matrix approximation schemes.
- Permutation can be arbitrary, excessive communication possible.
Low-Rank Approximations: Randomized Sampling

Algorithm RandSam0

- **Input:** mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix Ω.
2. Compute QR = A Ω
3. and SVD: $Q^T A = \hat{U} \hat{\Sigma} \hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- **Output:** $B = (Q \hat{U}_k) \hat{\Sigma}_k \hat{V}_k^T$
Low-Rank Approximation: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix $\Omega$.
2. Compute $QR = A \Omega$
3. and SVD: $Q^T A = \hat{U} \hat{\Sigma} \hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$

- Easy to implement.
- Very efficient computation.
- Minimum communication.
Low-Rank Approximation: Randomized Sampling

**Algorithm RandSam0**

- **Input**: mxn matrix A, int k, p.
  
  1. Draw a random nx(k+p) matrix Ω.
  2. Compute QR = A Ω
  3. and SVD: \( Q^T A = \hat{U} \hat{\Sigma} \hat{V}^T \)
  4. Truncate SVD: \( \hat{U}_k \hat{\Sigma}_k \hat{V}_k^T \)

- **Output**: \( B = (Q \hat{U}_k) \hat{\Sigma}_k \hat{V}_k^T \)

- Easy to implement.
- Very efficient computation.
- Minimum communication.

**Thm [(Remarkable) Limited Warranty]**

(Halko/Martinsson/Tropp, 2011) with failure probability \( 5p^{-p} \)

\[ \| A - B \|_2 = O(\sigma_{k+1}) \gg \sigma_{k+1} \]
Low-Rank Approximation: Randomized Sampling

**Algorithm RandSam0**

- **Input**: mxn matrix $A$, int $k$, $p$.

1. Draw a random nx$(k+p)$ matrix $\Omega$.
2. Compute $QR = A \Omega$.
3. and SVD: $Q^TA = \hat{U}\hat{\Sigma}\hat{V}^T$.
4. Truncate SVD: $\hat{U}_k\hat{\Sigma}_k\hat{V}_k^T$.

- **Output**: $B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$.

- Easy to implement.
- Very efficient computation.
- Minimum communication.

**Thm** [(Remarkable) Limited Warranty](Halko/Martinsson/Tropp, 2011)

with failure probability $5p^{-p}$

$$\|A - B\|_2 = O(\sigma_{k+1}) \gg \sigma_{k+1}$$

4 lines of code
40 pages of analysis
Low-Rank Approximation: Randomized Sampling

Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix $\Omega$.
2. Compute QR = A $\Omega$
3. and SVD: $Q^T A = \hat{U} \hat{\Sigma} \hat{V}^T$
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$

- Output: $B = (Q \hat{U}_k) \hat{\Sigma}_k \hat{V}_k^T$

- Easy to implement.
- Very efficient computation.
- Minimum communication.

Thm [(Remarkable) Limited Warranty] (Halko/Martinsson/Tropp, 2011)

\[ \| A - B \|_2 = O(\sigma_{k+1}) \gg \sigma_{k+1} \]

with failure probability $5p^{-p}$

(For $p = 13$, $5p^{-p} = 1.6 \times 10^{-14}$)
Is $10^{-14}$ small enough failure chance?

Chance of DNA match $= 10^{-14}$
Is $10^{-14}$ small enough failure chance?

Chance of DNA match = $10^{-14}$ DNA Can fail
For the Truly Motivated, Bound in Full Glory

\[ \| A - B \|_2 \leq \sigma_{k+1} + \| (I - P_Y)A \|_2 \]

**Theorem 10.8** (Deviation bounds for the spectral error). *Frame the hypotheses of Theorem 10.5. Assume further that \( p \geq 4 \). For all \( u, t \geq 1 \),

\[ \| (I - P_Y)A \| \leq \left[ \left( 1 + t \cdot \sqrt{12k/p} \right) \sigma_{k+1} + t \cdot \frac{e^{\sqrt{k+p}}}{p+1} \left( \sum_{j>k} \sigma_j^2 \right)^{1/2} \right] + ut \cdot \frac{e^{\sqrt{k+p}}}{p+1} \sigma_{k+1}, \]

with failure probability at most \( 5t^{-p} + e^{-u^2/2} \).
Low-Rank Matrix Approximations: Randomized Sampling

**Algorithm RandSam0**

- Input: mxn matrix A, int k, p.
  1. Draw a random nx(k+p) matrix Ω.
  2. Compute QR = A Ω
  3. and SVD: \( Q^T A = \hat{U} \hat{\Sigma} \hat{V}^T \)
  4. Truncate SVD: \( \hat{U}_k \hat{\Sigma}_k \hat{V}_k^T \)

- Output: \( B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T \)

- Earlier work by Rokhlin/Tygert (2008)
- Many variations.
- Randomized rank-revealing QR (Demmel/Dumitriu/Holtz, 2008; Toledo, 2010)
Algorithm RandSam0

- Input: mxn matrix A, int k, p.

1. Draw a random nx(k+p) matrix $\Omega$.
2. Compute QR = $A \Omega$
3. and SVD: $Q^TA = \hat{U}\hat{\Sigma}\hat{V}^T$
4. Truncate SVD: $\hat{U}_k\hat{\Sigma}_k\hat{V}_k^T$

- Output: $B = (Q\hat{U}_k)\hat{\Sigma}_k\hat{V}_k^T$

- Algorithm can work far better than theory predicts.
Improved Randomized Sampling

Algorithm RandSam1

- **Input:** mxn matrix $A$, int $k$, $p$, $c$.

1. Draw a random nx$(k+p+c)$ matrix $\Omega$.
2. Compute $QR = A\,\Omega$.
3. and SVD: $Q^T A = \hat{U} \hat{\Sigma} \hat{V}^T$.
4. Truncate SVD: $\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$.

- **Output:** $B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$.

Only change from RandSam0: $p$ becomes $p + c$.

Smallest modification of any algorithm.
Improved Randomized Sampling

Algorithm RandSam1

- **Input:** mxn matrix A, int k, p, c.
- **Output:**

1. Draw a random nx(k+p+c) matrix $\Omega$.
2. Compute $QR = A \Omega$
3. and SVD: $Q^T A = \hat{U} \Sigma \hat{V}^T$
4. Truncate SVD: $\hat{U}_k \Sigma_k \hat{V}_k^T$

$B = (Q\hat{U}_k) \Sigma_k \hat{V}_k^T$

Only change from RandSam0: p becomes $p + c$

Smallest modification of any algorithm.

c allows a drastically different error bound, controls accuracy.

p remains in control of failure chance.
Improved Randomized Sampling

$A = 2000 \times 2000$ random matrix with geometrically decaying singular values.

$p = 10$, $k = 20$. $\sigma_1 = 45$, $\sigma_{20} = 2$

Accuracy increases with $c$
Algorithm RandSam2

Input: mxn matrix A, int k, p, c, q.

1. Draw a random nx(k+p+c) matrix $\Omega$.
2. Compute QR of $(AA^T)^q A \Omega$ and SVD:
3. Truncate SVD:
4. Output:

$$B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$$

QR needs done carefully for numerical accuracy.

Algorithm is old one when q = 0; but q = 1 far more accurate.

Should converge faster when singular values do not decay very fast.

Thm [Limited Warranty]
(Halko/Martinsson/Tropp, 2011)

$$\|A - B\|_2 = O(\sigma_{k+1}) > \sigma_{k+1}$$

with failure probability $5p^{-p}$
Algorithm RandSam2

- Input: mxn matrix A, int k, p, c, q.

1. Draw a random nx(k+p+c) matrix $\Omega$.
2. Compute QR of $(AA^T)^q A \Omega$.
3. and SVD:
4. Truncate SVD:

$$Q^T A = \hat{U} \hat{\Sigma} \hat{V}^T$$
$$\hat{U}_k \hat{\Sigma}_k \hat{V}_k^T$$

- Output:

$$B = (Q\hat{U}_k)\hat{\Sigma}_k \hat{V}_k^T$$

QR needs done carefully for numerical accuracy.
Algorithm is old one when $q = 0$; but $q = 1$ far more accurate.
Should converge faster when singular values do not decay very fast.

Thm [Limited Warranty]
(Halko/Martinsson/Tropp, 2011)

$$\|A - B\|_2 = O(\sigma_{k+1}) > \sigma_{k+1}$$
with failure probability $5p^{-p}$

Bounds both stronger and weaker than those for traditional Subspace Iteration

Traditional Subspace Iteration with random start matrix
New Error Bound Analysis

**Theorem 5.8.** Let $A = U \Sigma V^T$ be the SVD of $A$, and $0 \leq p \leq \ell - k$. Further let $Q B_k$ be a rank-$k$ approximation computed by Algorithm 2.2. Given any $0 < \Delta \ll 1$, define

$$C_\Delta = \frac{e^{\sqrt{\ell}}}{p+1} \left( \frac{2}{\Delta} \right)^{\frac{1}{p+1}} \left( \sqrt{n - \ell + p} + \sqrt{\ell} + \sqrt{2 \log \frac{2}{\Delta}} \right).$$

We must have for $j = 1, \ldots, k$,

$$\sigma_j(QB_k) \geq \frac{\sigma_j}{\sqrt{1 + C_\Delta^2 \left( \frac{\sigma_{\ell-p+1}}{\sigma_j} \right)^{4q+2}}}.$$

and

$$\| (I - QQ^T) A \|_F \leq \| A - Q B_k \|_F \leq \sqrt{\left( \sum_{j=k+1}^{n} \sigma_j^2 \right) + k C_\Delta^2 \sigma_{\ell-p+1}^2 \left( \frac{\sigma_{\ell-p+1}}{\sigma_k} \right)^{4q}}$$

$$\| (I - QQ^T) A \|_2 \leq \| A - Q B_k \|_2 \leq \sqrt{\sigma_{k+1}^2 + k C_\Delta^2 \sigma_{\ell-p+1}^2 \left( \frac{\sigma_{\ell-p+1}}{\sigma_k} \right)^{4q}}.$$

with exception probability at most $\Delta$. 
Fast Randomized Algorithm with Subsampled random Fourier Transform: $T_{\text{struct}} \sim mn \log(\ell) + \ell^2 n$

HALKO, MARTINSSON, AND TROPP

Algorithm 4.5: Fast Randomized Range Finder

Given an $m \times n$ matrix $A$, and an integer $\ell$, this scheme computes an $m \times \ell$ orthonormal matrix $Q$ whose range approximates the range of $A$.

1. Draw an $n \times \ell$ SRFT test matrix $\Omega$, as defined by (4.6).
2. Form the $m \times \ell$ matrix $Y = A\Omega$ using a (subsampled) FFT.
3. Construct an $m \times \ell$ matrix $Q$ whose columns form an orthonormal basis for the range of $Y$, e.g., using the QR factorization $Y = QR$. 

26
Numerical Experiment (I): Computing truncated SVD

Comparison between randomized algorithm and svds

\[ A = (\log |x_i - y_j|) \]

\( x_i, y_j \) disjoint 2D points

is 4000x4000 matrix,

Numbers of Matrix-Vector Multiples

<table>
<thead>
<tr>
<th>Tolerance</th>
<th>( q = 0 )</th>
<th>( q = 2 )</th>
<th>( q = 4 )</th>
<th>svds</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 10^{-6} )</td>
<td>143</td>
<td>5 \times 96</td>
<td>9 \times 79</td>
<td>500</td>
</tr>
<tr>
<td>( 10^{-8} )</td>
<td>180</td>
<td>5 \times 96</td>
<td>9 \times 87</td>
<td>600</td>
</tr>
<tr>
<td>( 10^{-10} )</td>
<td>190</td>
<td>5 \times 96</td>
<td>9 \times 93</td>
<td>600</td>
</tr>
</tbody>
</table>
Numerical Experiment (II):
Fast Structured Matrix Preconditioners:

T. Davis' SPD Sparse Matrix

A = Bottom Schur Complement of dimension 3300.
CG takes 878 iterations for $10^{-12}$ residual

\[ \begin{array}{l}
\text{Numbers of PCG Iterations} \\
\hline
\text{Maximum off-diagonal rank } k & p = 10 & p = 20 & p = 40 \\
\hline
20 & 75 & 77 & 72 \\
40 & 69 & 69 & 69 \\
60 & 64 & 61 & 61 \\
\end{array} \]
Numerical Experiment (III): Eigenface
Comparison with truncated SVD for 200 classifications

Comparison of Numbers of Incorrect Matches

<table>
<thead>
<tr>
<th>Rank $k$</th>
<th>$p = 10$</th>
<th>$p = 20$</th>
<th>$p = 40$</th>
<th>Truncated SVD</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>32</td>
<td>25</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
<td>26</td>
<td>25</td>
<td>21</td>
</tr>
<tr>
<td>30</td>
<td>21</td>
<td>20</td>
<td>18</td>
<td>17</td>
</tr>
<tr>
<td>40</td>
<td>20</td>
<td>17</td>
<td>17</td>
<td>16</td>
</tr>
</tbody>
</table>
Numerical Experiment (IV): QRCP vs. Randomized Algorithm

\[ N = 4000; \]
\[ x = (\pi/2)*(-N:N)/N; \]
\[ e = \text{ones}(2*N+1,1); \]
\[ B = \log((\text{abs}(\sin(e*x-x'*e')))); \]
\[ B(\sim \text{isfinite}(B)) = 0; \]
\[ B=B(1:N/2,N/2+1:end); \]

\[ p = 30; \]
\[ W = \text{randn}(N+N/2+1,p); \]
\[ BW = B*W; \]
\[ [QW,RW]=\text{qr}(BW,0); \]
\[ [UW,SW,VW] = \text{svd}(B'*QW,0); \]
\[ srnd = \text{diag}(SW); \]

\[ [QB,RB,P] = \text{qr}(B,0); \]
\[ sqr = \text{svd}(RB(1:p,:)); \]
Numerical Experiment (IV): QRCP vs. Randomized Algorithm

Approx Svalues by Random Alg, QRCP

- Rnd Svalues
- QR Svalues
Numerical Experiment (IV): QRCP vs. Randomized Algorithm

Svalues Diffs by Random Alg, QRCP

10^{-14} 10^{-13} 10^{-12} 10^{-11} 10^{-10} 10^{-9} 10^{-8} 10^{-7} 10^{-6}

0 5 10 15 20 25 30
Numerical Experiment (IV): QRCP vs. Randomized Algorithm

![Graph showing S-values errors by Random Alg](image)
Current Work On Randomized Algorithms

- Randomized Gaussian elimination with complete pivoting

- Randomized spectrum-revealing LU, QR, Cholesky factorizations
  (vs. CUR/CX decompositions, pivoted Cholesky factorizations, randomized low-rank matrix approximations)

- Big Data applications.